EA extension | for Quality Assurance

User Guide

Versión 2.0

deiser | Xpress Learning

1. INTRODUCTION 2

2. TRACEABILITY FUNCTIONS 3

RULES MANAGER 3

CHECK 5

ARTIFACT TRACEABILITY 7

3. STATISTICAL FUNCTIONS 10

ARTIFACTS COUNT 10

CLASS OPERATIONS AVERAGE 11

USE CASE SCENARIOS AVERAGE 11

4. MASSIVE UPDATE FUNCTIONS 12

PROPERTY / TAGGED VALUE MODIFICATIONS 12

ATTRIBUTE / COLUMN DATA TYPE MODIFICATIONS 13

5. CHECK AND VERIFICATIONS FUNCTIONS 14

ARTIFACTS WITHOUT DESCRIPTION 14

NOMENCLATURE CHECK 15

SEARCH FOR CYCLICAL RELATIONSHIPS 17

SEARCH FOR MULTIPLE INHERITANCE 19

SEARCH FOR REPEATING RELATIONSHIPS 20

ARTIFACT x PACKAGE MATRIX GENERATION 22

USE CASE VERIFICATION 24

6. OTHER FUNCTIONS 26

CLONE EMELENT 26

DISCONNECT BRANCH FOR VERSION CONTROL 27

CHANGE LANGUAGE 28

ADD LICENSE NUMBER 28

ABOUT... 29

1. INTRODUCTION

The aim of this guide is to explain in detail the user operational point of view of ***EA Extension for QA*** to function properly.

To that end, all the plugin's functionalities will be described in this document.

2. TRACEABILITY FUNCTIONS

RULES MANAGER

This functionality is aimed to let users to design the collection of traceability rules which be applied in the check process (explained later).

These rules will let define the set of regulations about necessary traceability in certain environments. The definition of each rule consists in the elaboration of statements close to these:

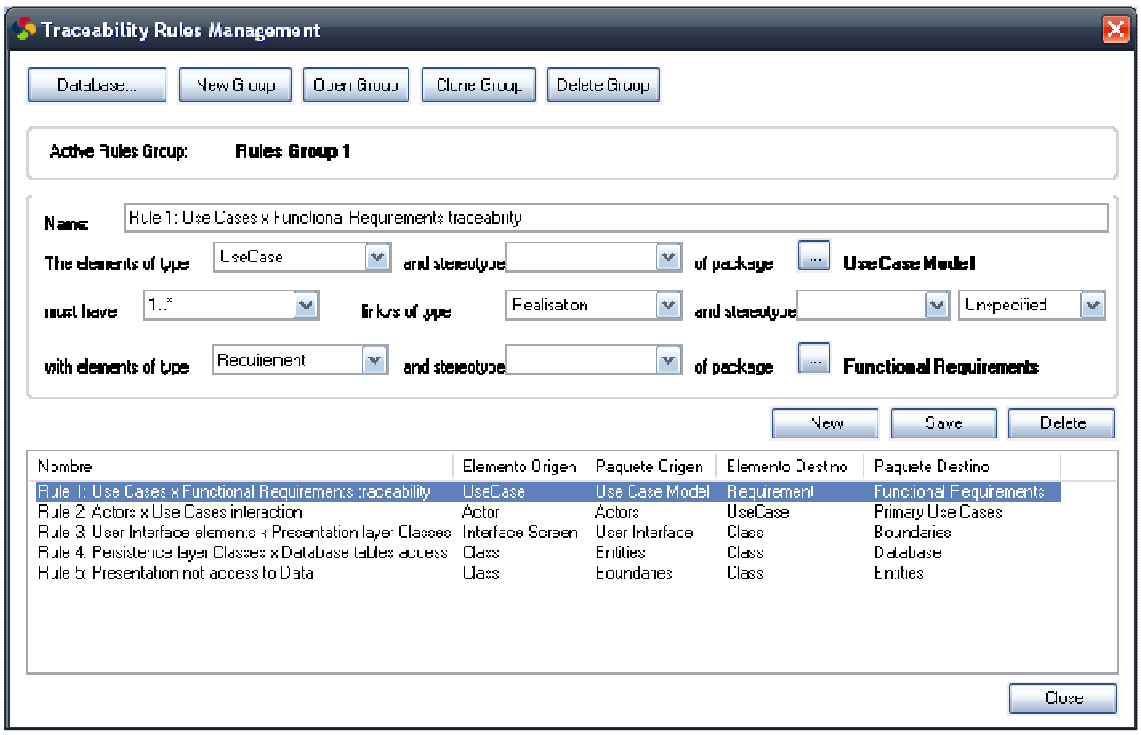
"Every Use Cases have to had one or several relationships with Functional Requirements"

"Every Screen Prototypes have to had only one relationship with boundary Classes"

"Every Components have to had a deploy relationship with Nodes"

"Every Actors have to had one or several Use relationships with Use Cases"

The Rules Composer's interface is as shown in the image:



The buttons and main areas are described below:

Database...: Let setup what database will be used in order to manage the Rules. This database may be shared by several users if it is located, for example, on a network. Traceability Rules' database will be stored in Microsoft Access format (.mdb)

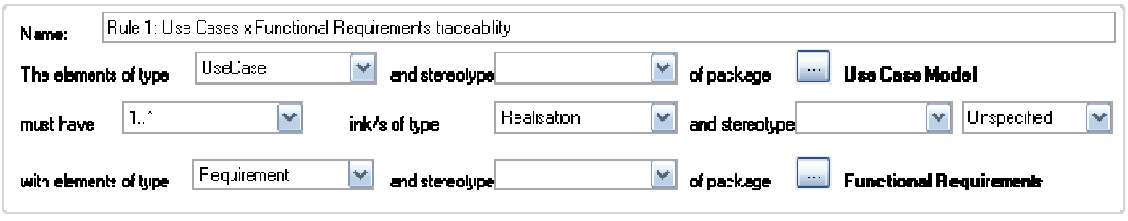
New Group: When the database is selected, it is possible, using this button, to create New Groups of Rules. The Rules' Groups contains Traceability Rules and they are useful to make easier their management and to apply them to different typologies' Project, sizes, etc.

Open Group: When the database is selected, it is possible, using this button, to open (to enable) Groups of Rules. Once a Group of Rules is enabled, it will be able to work with the Rules contained in it (doing modifications, erasing rules or creating new ones).

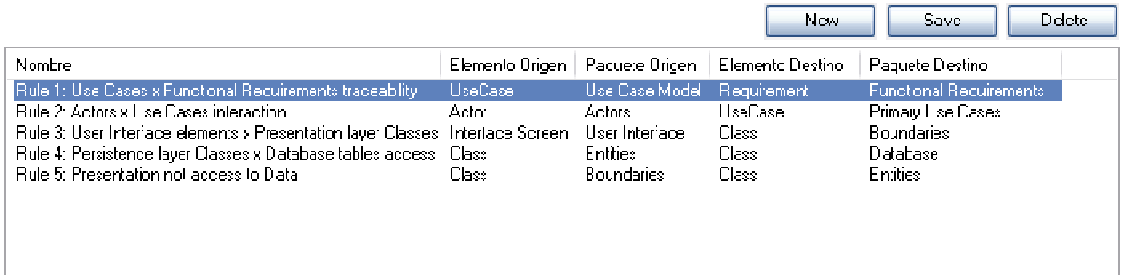
Clone Group: This button let to clone the active Group of Rules, so it is possible to create a new Group of Rules from an existing one in order to, later, do the appropriate modifications.

Delete Group: This button erases the active Group of Rules of the Rules' Database.

Active Rules Group: Shows the Group of Rules active at every moment.



Traceability Rule's Definition Zone: it lets users to setup the statement of each Traceability Rule. It is established the type and stereotype of source elements, the container Package of source elements, the number of relationships that have to exist between source and target elements, the type, stereotype and navigability of the relationships that have to exist, the type and stereotype of the target elements and the container Package of target elements.

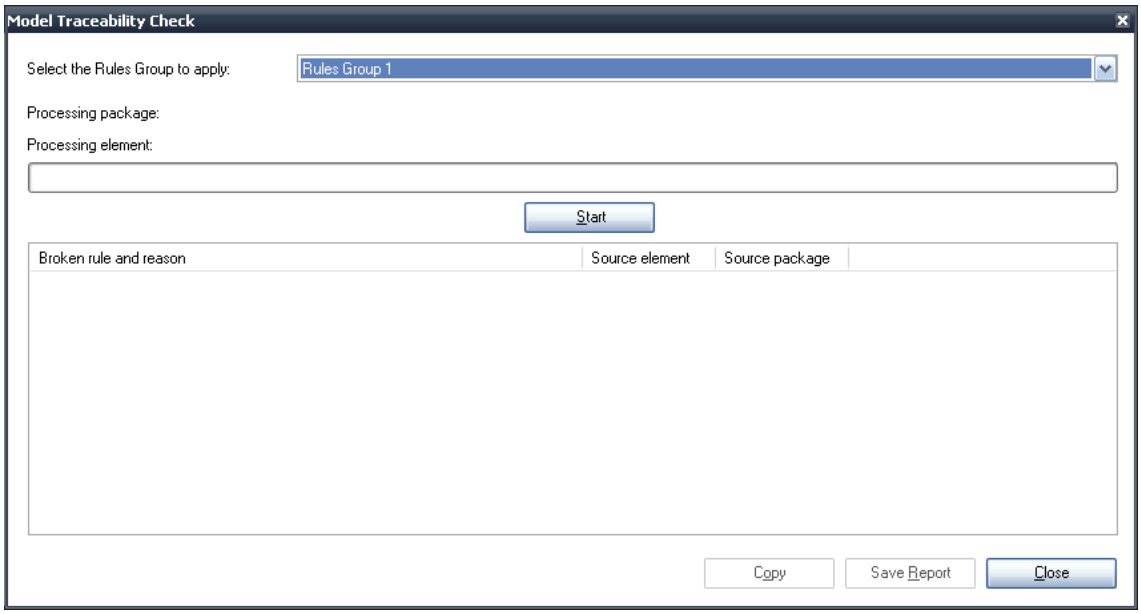


New: Defines a new Traceability Rule in the active Group of Rules
Save: Saves the definition of the Traceability Rule in the active Group of Rules.
Delete: Erases the selected Traceability Rule from the active Group of Rules.
Rules List: Shows all the Traceability Rules from the active Group of Rules.

CHECK

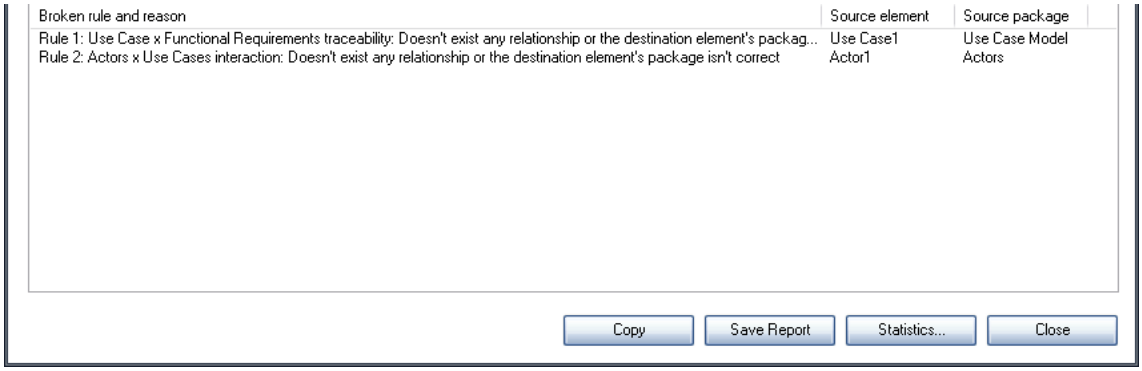
This functionality lets to apply a Collection of Traceability Rules to a selected Package.

The interface is as shown in the image:



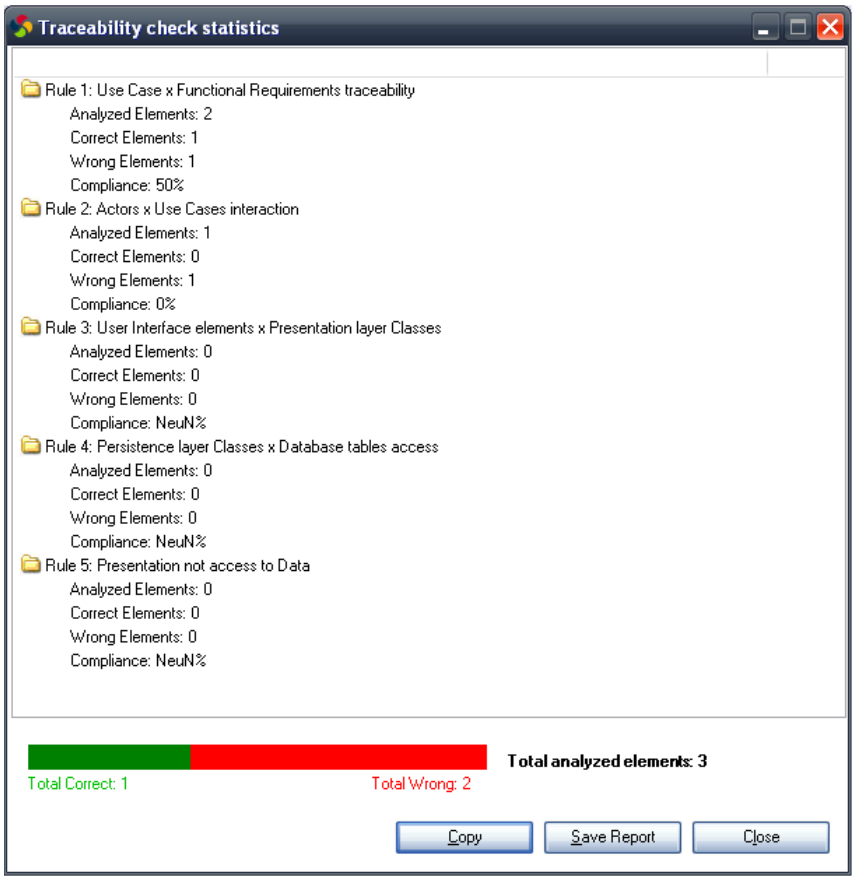
User has to select the Group of Rules to apply during the verification process.

Once the button "Start" is clicked, the check process starts applying the selected Group of Rules. When the process ends, the elements which do not fulfil with the established Traceability Rules will be displayed:



Finally, the interface provides the following options for the results:

Copy: Copies the results to Windows' Clipboard.
Save Report: Exports the reports to a text files.
Close: Closes Model Check interface.
Statistics...: Shows a window with the check statistics for each Rule.



In the "Treaceability check statistics" window, appears the following buttons:

Copy: Copy the statistics to the Windows clipboard.
Save Report: Export the results to a CSV file.
Close: Closes the check statistics window.

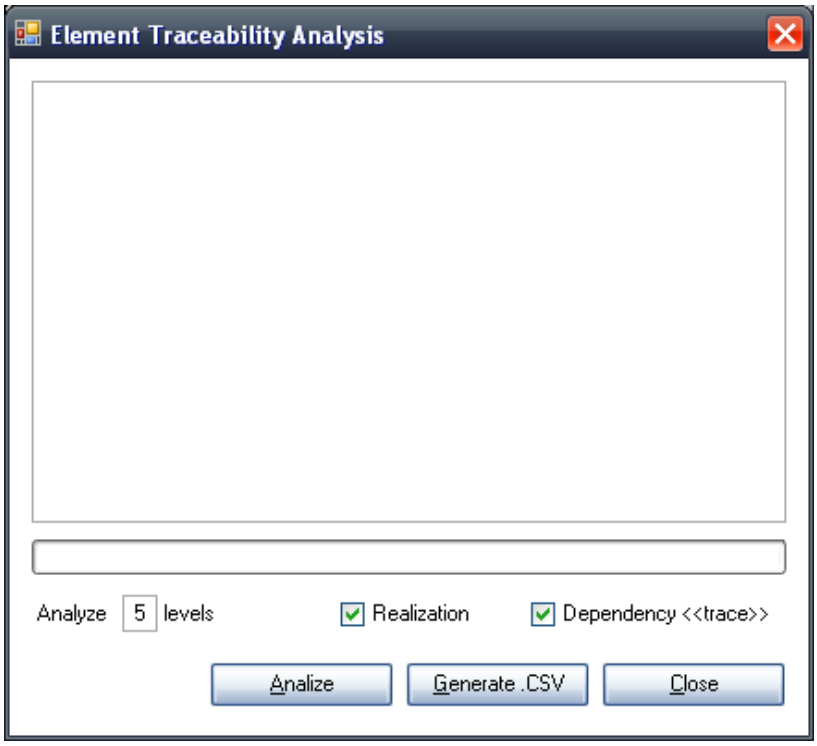
ARTIFACT TRACEABILITY

This functionality applies to the Elements of the Repository and allow to users to examine the traceability of those elements, generating a traceability tree which can be exported to a .CSV file.

This functionality is useful, for example, when it is done an impact analysis on a specific element.

Enterprise Architect has an interface which realises a similar function (View | Traceability) but it is not possible to export the analysis to a file. Moreover, the analysis performed by Enterprise Architect only can be done for 5 levels in depth and it is not displayed the stereotype of the reported elements.

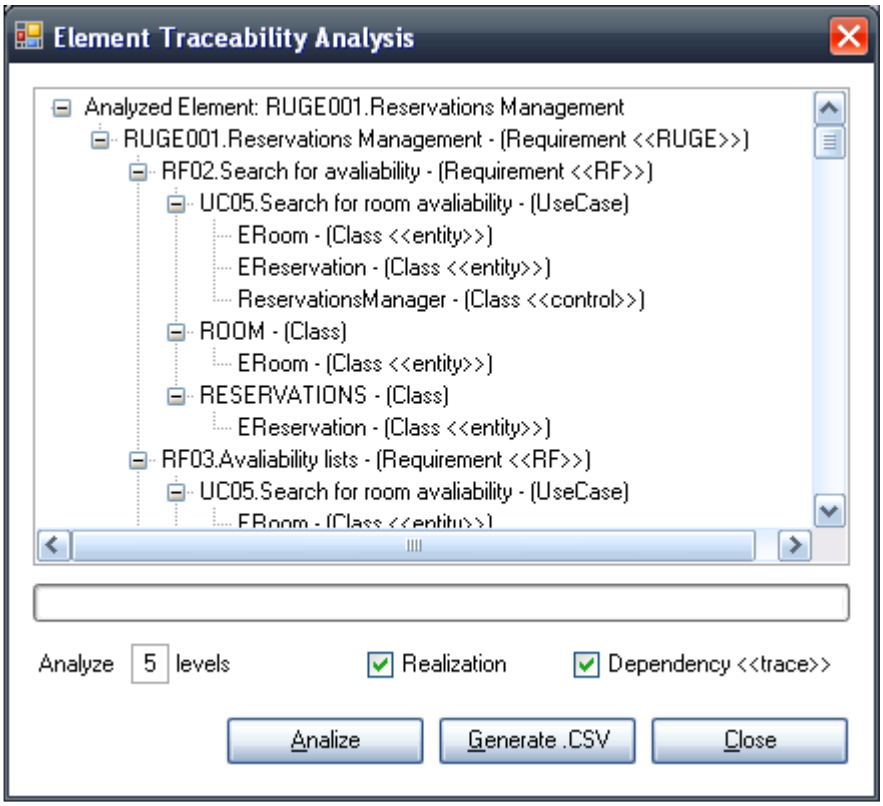
The interface about element traceability analysis is as shown in the image:



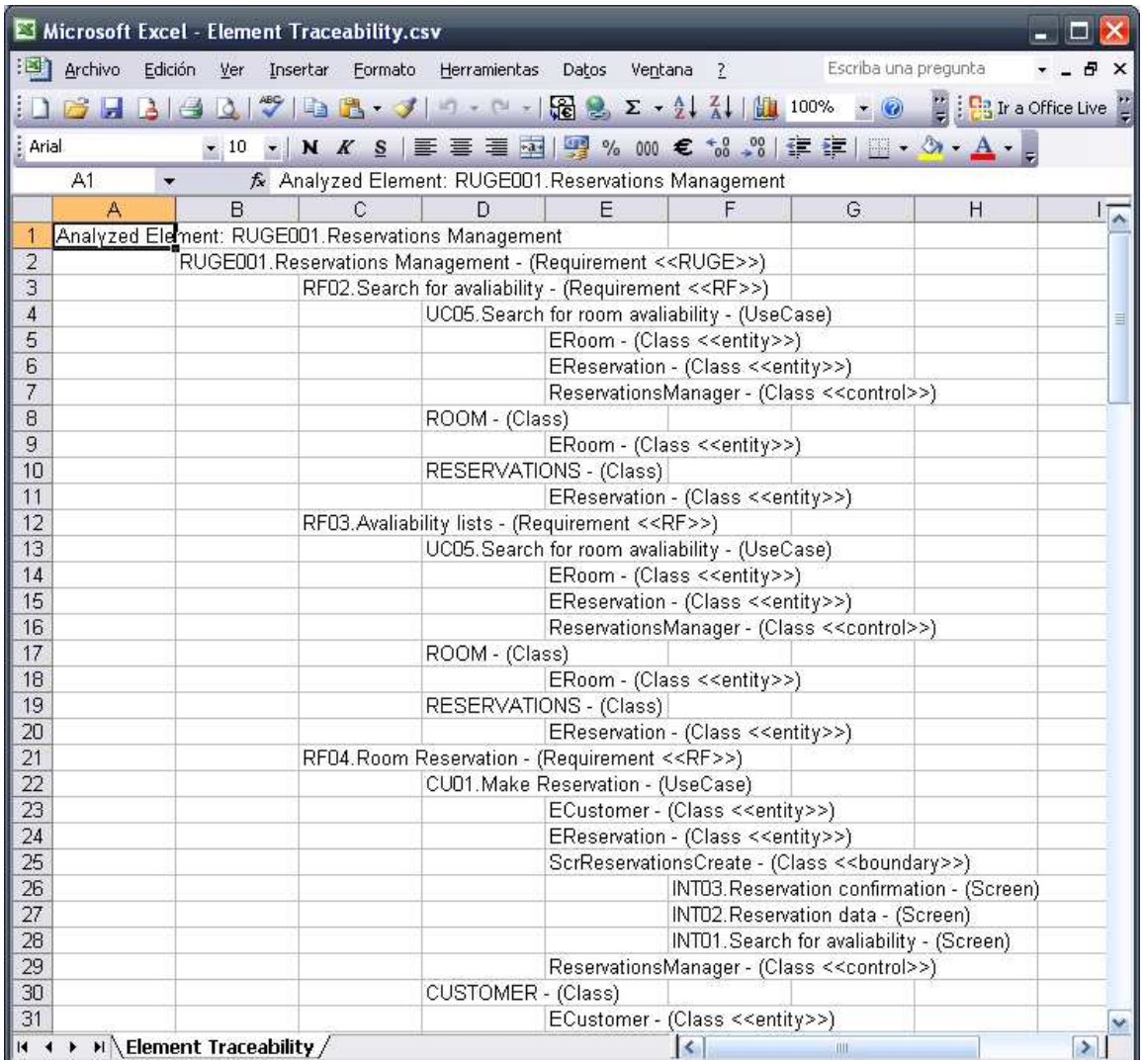
It is possible to define up to 9 levels in depth as well as to define what relationships will be taken into account in the execution of the process (**Realization** and/or **Dependency** with trace stereotype).

When button "Analyze" is clicked, the process will start and it may take just a few seconds or several minutes, depending on the complexity on the established relationships in the analyzed element and the depth selected.

Once the analysis process is finished, it is shown the traceability tree of the analyzed element.



By means of "Generate .CSV" button it is possible to export to a file the content of the tree as is shown in the next image:



	A	B	C	D	E	F	G	H	I
1	Analyzed Element: RUGE001.Reservations Management								
2		RUGE001.Reservations Management - (Requirement <<RUGE>>)							
3			RF02.Search for avaliability - (Requirement <<RF>>)						
4				UC05.Search for room avaliability - (UseCase)					
5					ERoom - (Class <<entity>>)				
6					EReservation - (Class <<entity>>)				
7					ReservationsManager - (Class <<control>>)				
8				ROOM - (Class)					
9					ERoom - (Class <<entity>>)				
10				RESERVATIONS - (Class)					
11					EReservation - (Class <<entity>>)				
12			RF03.Avaliability lists - (Requirement <<RF>>)						
13				UC05.Search for room avaliability - (UseCase)					
14					ERoom - (Class <<entity>>)				
15					EReservation - (Class <<entity>>)				
16					ReservationsManager - (Class <<control>>)				
17				ROOM - (Class)					
18					ERoom - (Class <<entity>>)				
19				RESERVATIONS - (Class)					
20					EReservation - (Class <<entity>>)				
21			RF04.Room Reservation - (Requirement <<RF>>)						
22				CU01.Make Reservation - (UseCase)					
23					ECustomer - (Class <<entity>>)				
24					EReservation - (Class <<entity>>)				
25					ScrReservationsCreate - (Class <<boundary>>)				
26						INT03.Reservation confirmation - (Screen)			
27						INT02.Reservation data - (Screen)			
28						INT01.Search for avaliability - (Screen)			
29					ReservationsManager - (Class <<control>>)				
30				CUSTOMER - (Class)					
31					ECustomer - (Class <<entity>>)				

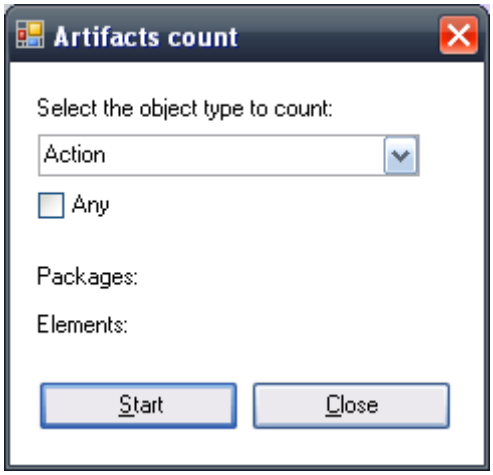
3. STATISTICAL FUNCTIONS

ARTIFACTS COUNT

This functionality applies to a Package and all of its subpackages.

It provides a simple count of the specified type of Elements in the selected Package (and all its subpackages)

The interface is as shown in the image:



The interface allows users to define:

- The filter which will be applied to the elements which are going to be processed (or, select all, if user marks the "Any" checkbox).

When the process is finished, the interface displays:

- The number of Packages founded Turing the count process
- The number of founded Elements (which fulfil with the filter selected)

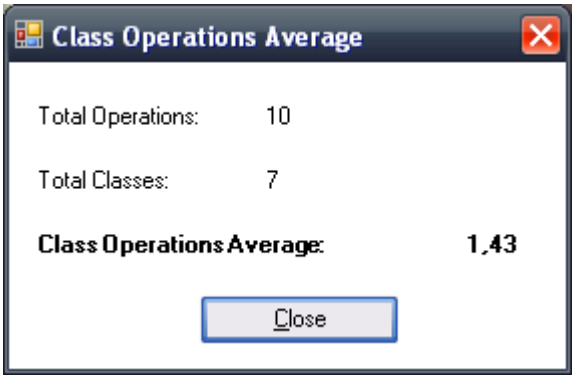
In order to start the count process the button "Start" have to be clicked.

CLASS OPERATIONS AVERAGE

This functionality applies to a Package and all of its subpackages

It provides the average of operations per class. To obtain this value, all the operations of the selected Package (and subpackages) of the Classes are processed.

Once the process is finished, this interface will be displayed:

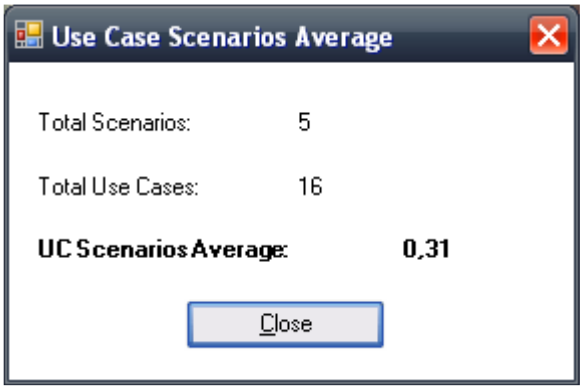


USE CASE SCENARIOS AVERAGE

This functionality applies to a Package and all of its subpackages

It provides the average of scenarios per use case. To obtain this value, all the scenarios of the selected Package (and subpackages) of the Use Cases are processed.

Once the process is finished, this interface will be displayed:



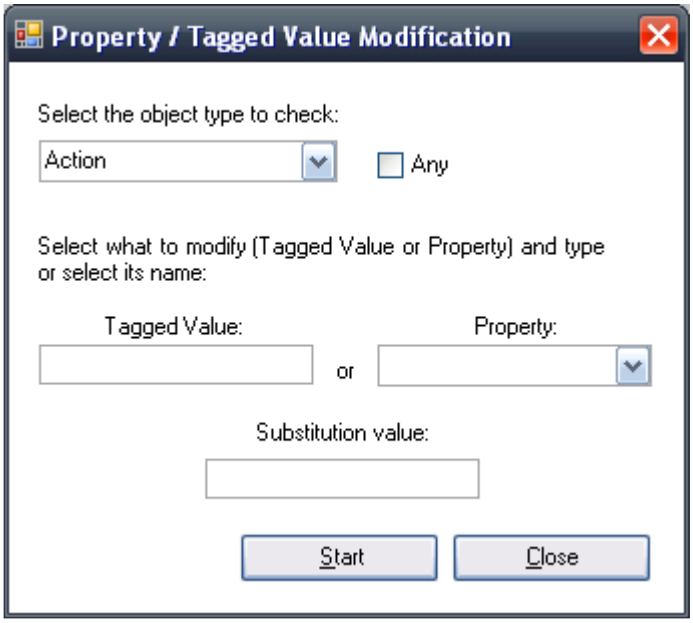
4. MASSIVE UPDATE FUNCTIONS

PROPERTY / TAGGED VALUE MODIFICATIONS

This functionality applies to a Package and all of its subpackages

It allows to users to modify the values of certain Properties and tagged Values in a massive way.

The interface is as shown in the image:



The interface allows users to define:

- The filter which will be applied to the elements which are going to be processed (or, select all, if user marks the "Any" checkbox).
- The name of the Tagged Value to modify.
- The name of the Property to modify.
- The substitution value for the Tagged Value or Property.

Note: only can be modified the Tagged Value or the Property but not both simultaneously.

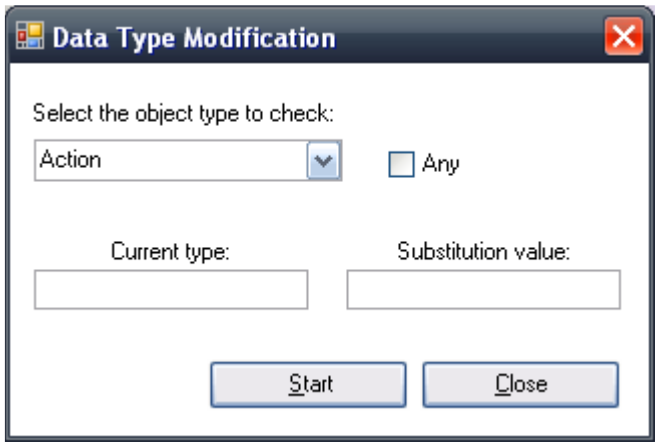
When the button "Start" is clicked, the modification process starts. Once it has finished, a notification is displayed to the user.

ATTRIBUTE / COLUMN DATA TYPE MODIFICATIONS

This functionality applies to a Package and all of its subpackages

It allow to users to modify the Data type of the Attributes (or Columns if elements are tables) of all elements of the filter.

The interface is as shown in the image:



The interface allows users to define:

- The filter which will be applied to the elements which are going to be processed (or, select all, if user marks the "Any" checkbox).
- The current data type that should be modified.
- The new data type that should be applied.

When button "Start" is clicked, the change process of data types of Attributes/Columns will start. This process changes the specified type in "Current type" box with the new type of "Substitution value" box. Once the process has finished, a notification is displayed to the user.

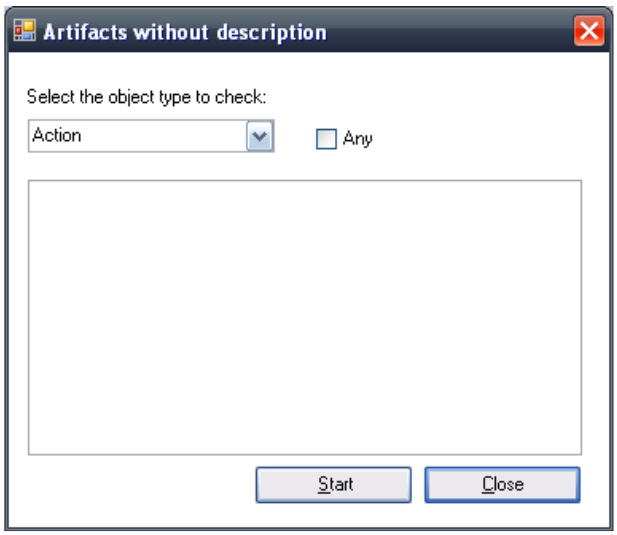
5. CHECK AND VERIFICATIONS FUNCTIONS

ARTIFACTS WITHOUT DESCRIPTION

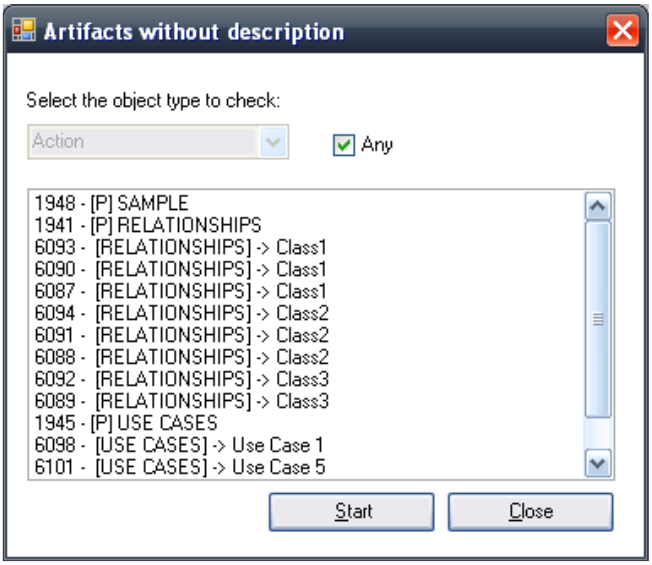
This functionality applies to a Package and all of its subpackages

Identifies those elements that have no description in the "Notes" field, from the elements set out in the filter.

The interface is as shown in the image:



By clicking the "Start" button starts the identification process, which ends showing a window like this:



For each element identified, is shown

6089 - [RELATIONSHIPS] -> Class3

Element ID – [Package where the item is located] - › Element Name

If you have set the filter to identify elements of type "Package" or have checked the box to "Any", also identify the packages with no description. In the results list will appear as follows

1945 - [P] USE CASES

Package ID – [P] Package name

[P]: indicates that is a Package.

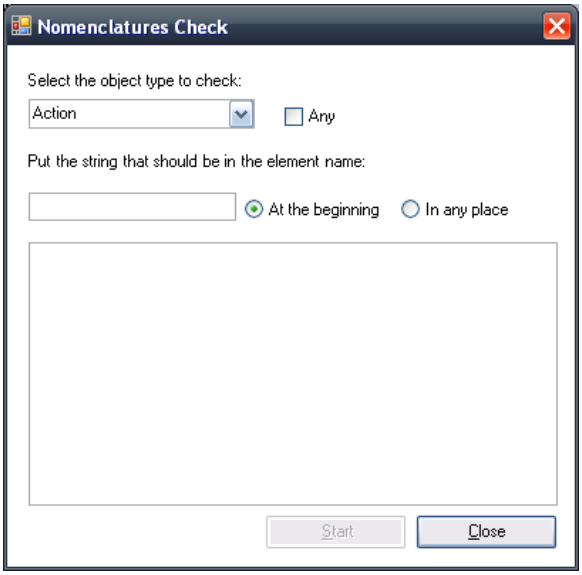
Double-clicking on any row of the list for a package or item identified, select the package or item in the Project Browser.

NOMENCLATURE CHECK

This functionality applies to a Package and all of its subpackages.

Identifies those elements that do not comply with the nomenclature established between the elements set out in the filter.

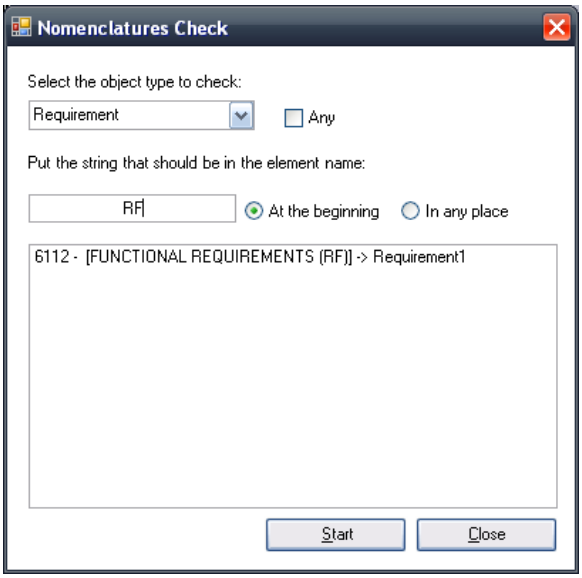
The interface is as shown in the image:



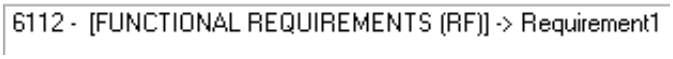
The interface allows users to define:

- Filter to apply the elements to be treated (or set by the box "Any" all elements are affected by the process).
- The text string that must be part of the name of the elements listed in the filter.
- If this string must appear "At the beginning," or "In any place" Item Name.

By clicking the "Start" button will start the process of identifying the elements in the filter that do not satisfy the requirement of nomenclature. After completing this process, the window will look like:



For each element identified, is shown:



Element ID – [Package where the item is located] - › Element Name

If you have set the filter to identify elements of type "Package" or have checked the box to "Any", also identify the packages with no description. In the results list will appear as follows:

Package ID – [P] Package name

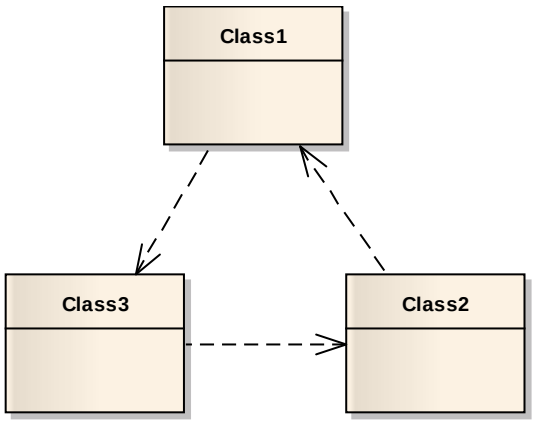
[P]: indicates that is a Package.

Double-clicking on any row of the list for a package or item identified, select the package or item in the Project Browser.

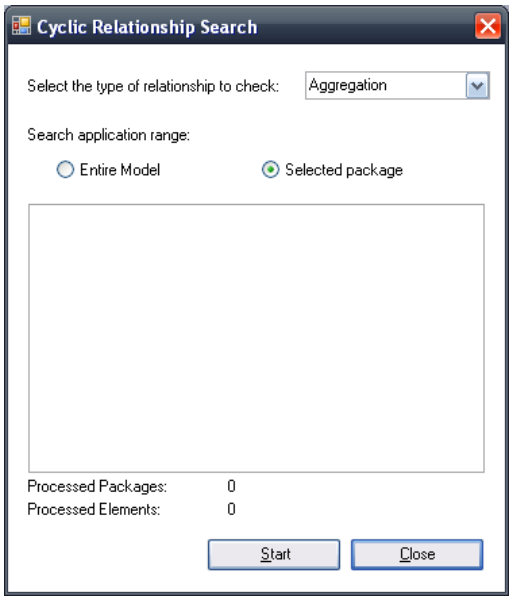
SEARCH FOR CYCLICAL RELATIONSHIPS

This functionality applies to a Package and all of its subpackages

Identifies those elements that contain a cyclic relationship on self through other. In short, is used to identify such situations:



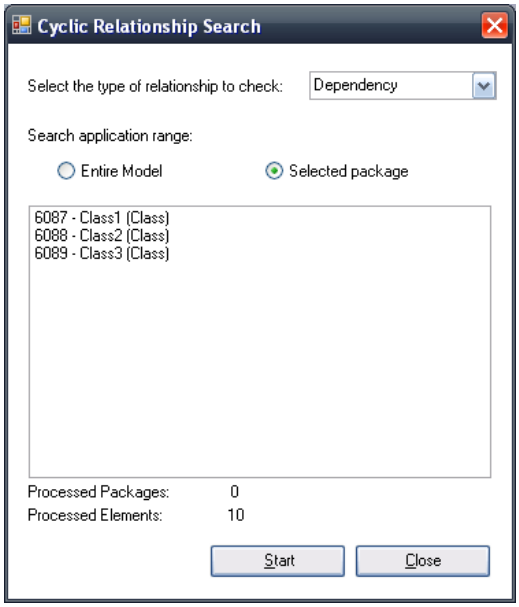
The interface is as shown in the image:



The interface allows users to define:

- The filter to apply to the type of relationship to be treated.
- If the search process cyclical relationship applies to "Any Model" (including all the root nodes that exist in the repository) or only to "selected package and its subpackages.

By clicking the "Start" button will start the search process cyclical relationships. After completing this process, the window will look like:



For each element identified cyclical relationships are shown:

6087 - Class1 (Class)

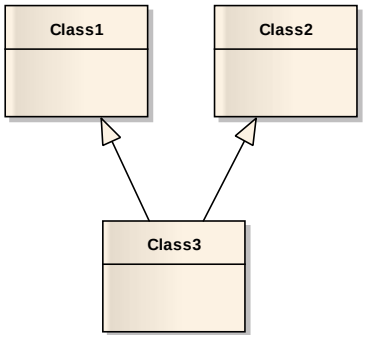
Element ID – Element name (Element type)

Double-clicking on any row of the list for an item identified, select the item in the Project Browser.

SEARCH FOR MULTIPLE INHERITANCE

This functionality applies to a Package and all of its subpackages.

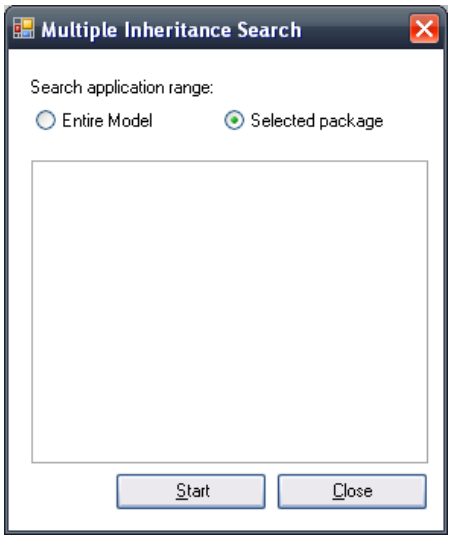
It can identify those classes that inherit from more than one class. In short, it serves to identify situations of this kind, not eligible for some programming languages:



The interface is as shown in the image, and allows users to define:

- If the process of searching for multiple inheritance applies to "Entire Model" (including all the root nodes that exist in the repository) or only to "Selected package" and its subpackages

By clicking the "Start" button will start the search process of multiple inheritance. After completing this process, the window will look like:



For each element found with multiple inheritance, shown:

6092 - Class3 (Class)

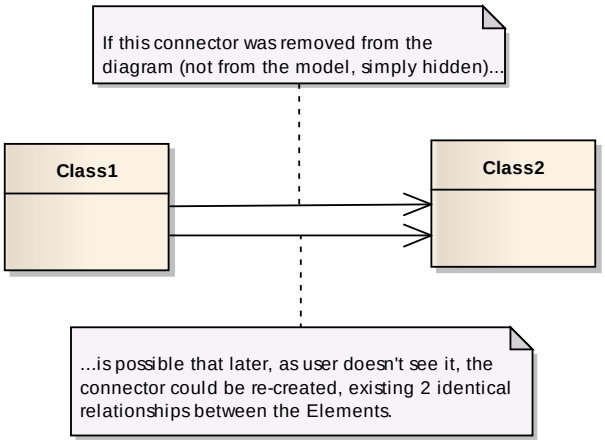
Element ID – Element name (Element type)

Double-clicking on any row of the list for an item identified, select the item in the Project Browser.

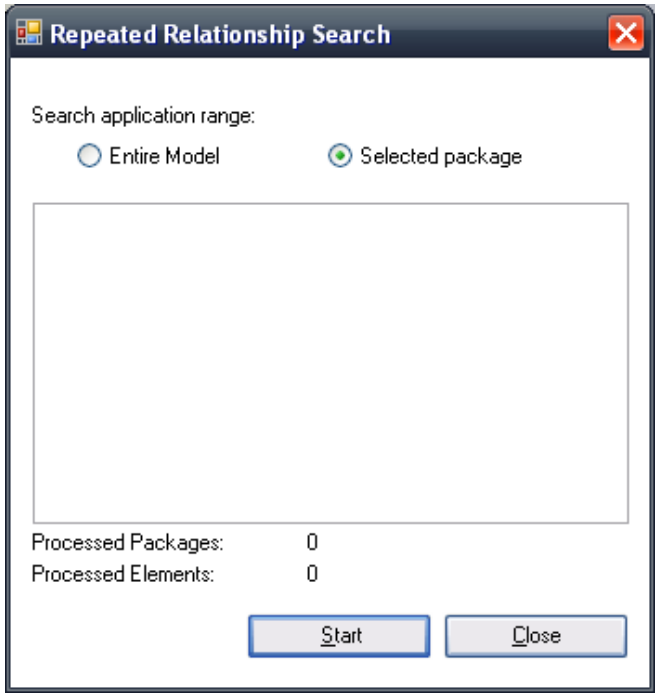
SEARCH FOR REPEATING RELATIONSHIPS

This functionality applies to a Package and all of its subpackages

Identifies those elements that contain a repeated relationship. This situation is especially common when, having hidden or deleted a connector in a diagram, then to not see it is created anew without verifying that, although not visible, the relationship already exists. In short, is used to identify such situations:



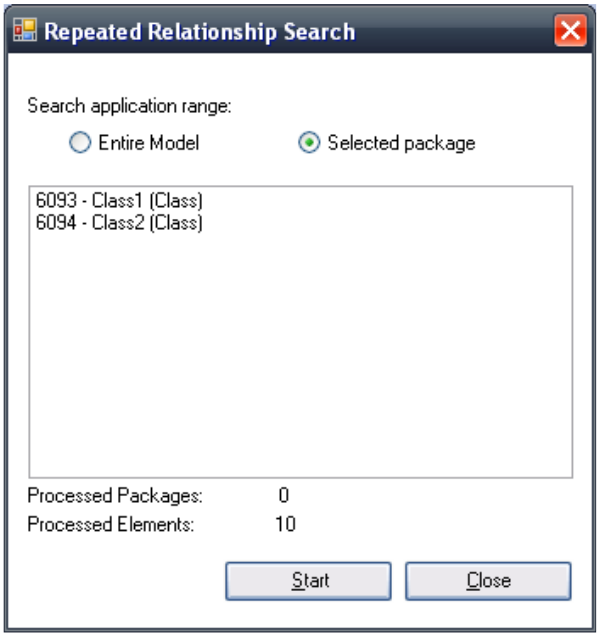
The interface looks like this:



The interface allows users to define:

- If the search process cyclical relationship applies to "Entire Model" (including all the root nodes that exist in the repository) or only to "Selected package" and its subpackages.

By clicking the "Start" button will start the search process repeated relationships. After completing this process, the window will look like:



For each element identified several relationships, it is shown:

6092 - Class3 (Class)

Element ID – Element name (Element type)

Double-clicking on any row of the list for an item identified, select the item in the Project Browser.

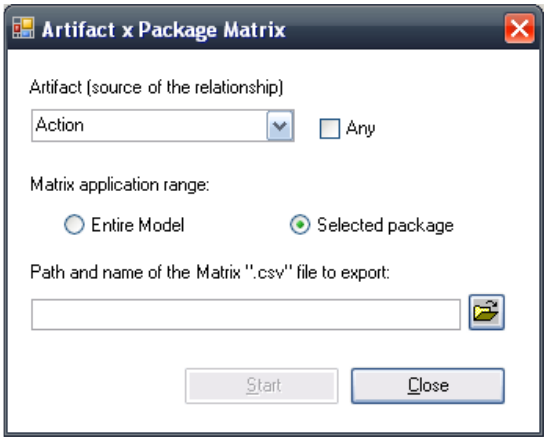
ARTIFACT x PACKAGE MATRIX GENERATION

This functionality applies to a Package and all of its subpackages

Enterprise Architect does not provide any default relationship between an element and the package that contains it. On the other hand, sometimes it is necessary to generate an matrix indicating which elements are determined Package

Therefore, this feature covers the deficiency which will generate a Matrix "structural" in which all elements are folded against the package in which they reside.

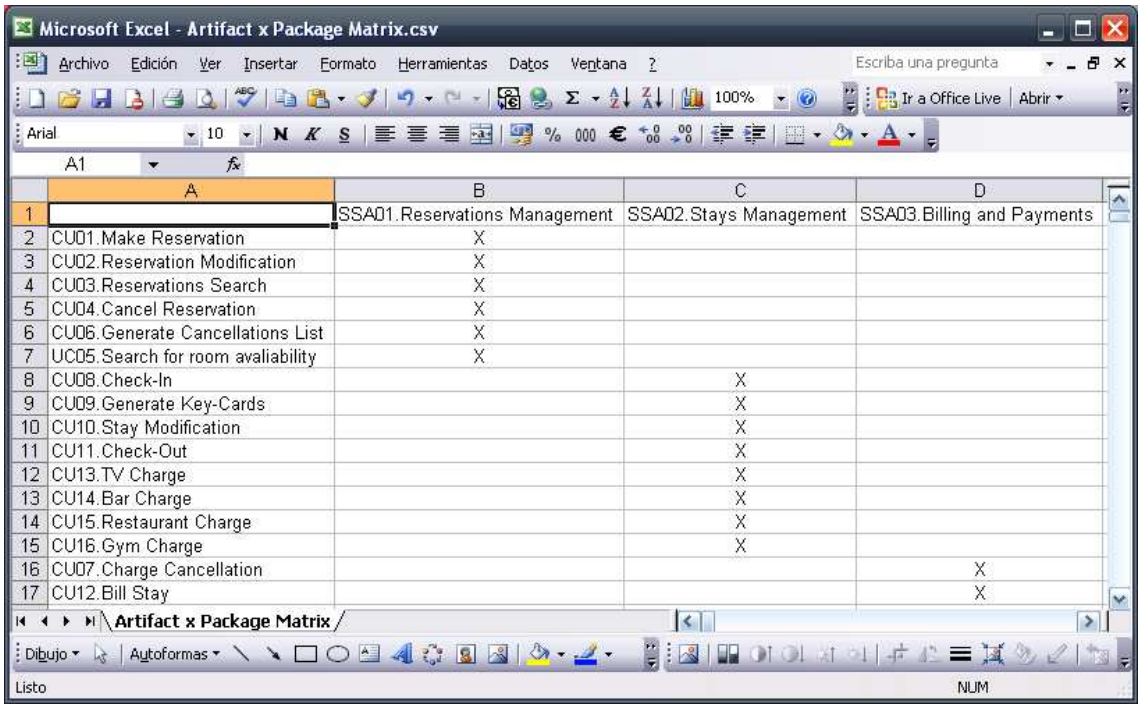
The interface is as shown in the image:



The interface allows users to define:

- The filter to apply the elements to be treated (or set by the box "Any" that ALL items will be affected by the process.)
- If the range of application of the Matrix applies to "Entire Model" (including all the root nodes that exist in the repository) or only to "Selected package" and its subpackages.
- The path and file name. CSV to be generated

By clicking the "Start" button will start the process of generation of the Matrix. After completing this process the generated .CSV file will look like:



	A	B	C	D
1		SSA01.Reservations Management	SSA02.Stays Management	SSA03.Billing and Payments
2	CU01.Make Reservation	X		
3	CU02.Reservation Modification	X		
4	CU03.Reservations Search	X		
5	CU04.Cancel Reservation	X		
6	CU06.Generate Cancellations List	X		
7	UC05.Search for room avaliability	X		
8	CU08.Check-In		X	
9	CU09.Generate Key-Cards		X	
10	CU10.Stay Modification		X	
11	CU11.Check-Out		X	
12	CU13.TV Charge		X	
13	CU14.Bar Charge		X	
14	CU15.Restaurant Charge		X	
15	CU16.Gym Charge		X	
16	CU07.Charge Cancellation			X
17	CU12.Bill Stay			X

USE CASE VERIFICATION

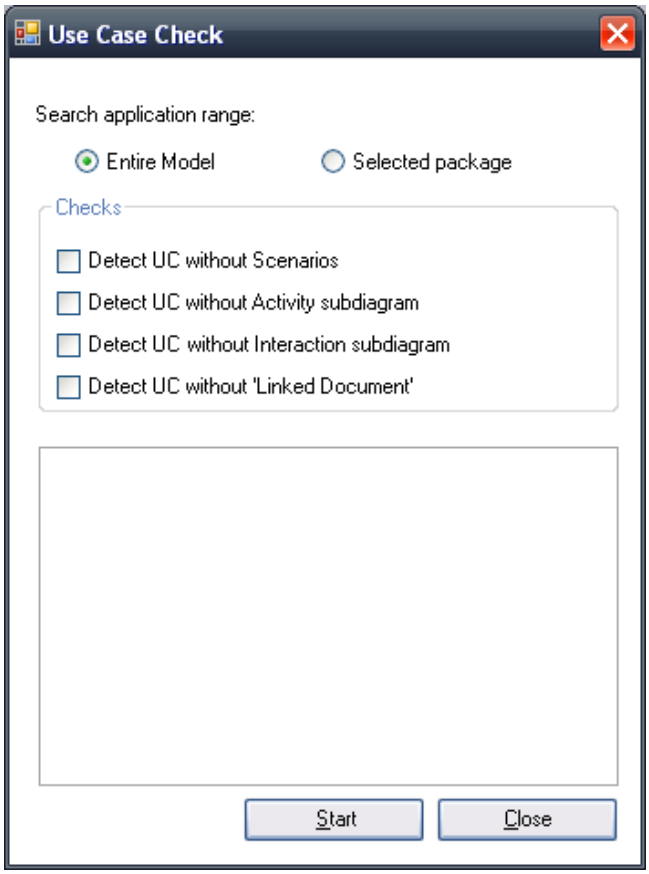
This functionality applies to a Package and all of its subpackages

Its functionality is to identify those cases which do not comply with the provides specification. It is very common for the development process requiring analysts to specify the Use Cases by one or more of the following techniques:

- Overview of the Use Case
- Textual description of each scenario
- Interaction diagrams that specify the performance of each scenario

Therefore, through this function to identify those cases which do not meet some or all of these needs

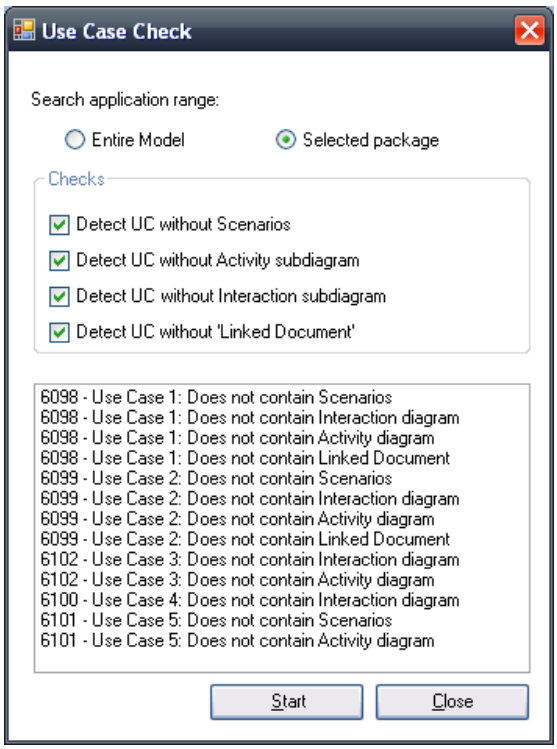
The interface is as shown in the image:



The interface allows users to define:

- If the application range of the search applies to "Entire Model" (including all root nodes that exist in the Repository) or only to "Selected package" and its subpackages.
- If you identify whether or not those cases which have not defined Scenarios in the "Scenarios" tab.
- If you identify whether or not those Use Cases which do not have any Activity subdiagram.
- If you identify whether or not those Use Cases which do not have any Interaction subdiagram (sequence or communicacion).
- If you identify whether or not those cases which do not have a "Linked Document" that describes its functionality.

By clicking the "Start" button will start the search process Use Cases that fail to comply with our guidelines. After completing this process, the window will look like:



For each element identified, it is shown:

6098 - Use Case 1: Does not contain Interaction diagram

Element ID – Element name: Breached Rule

Double-clicking on any row of the list for an item identified, select the item in the Project Browser.

6. OTHER FUNCTIONS

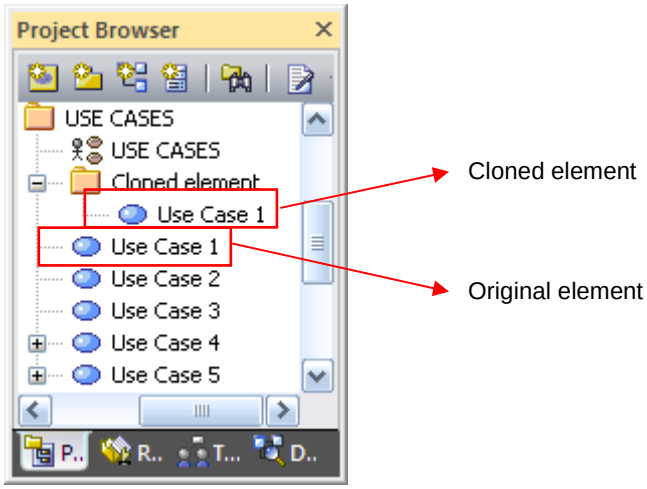
CLONE EMELENT

Although Enterprise Architect provides the functionality of "copy and paste" elements, this process only copies the element and its properties, but relations it has with its environment.

However, the functionality built into the supplement called "Clone Item", what it does internally is to make an export-import of the item so as to obtain an exact copy thereof, including relationships with other elements of the Repository.

This functionality is very useful when you want to create a very similar element to an existing, retaining the same relationships held by the original item.

To use this function, simply select the element to be cloned and use the menu "EA Extension for QA | Clone Element". After the process, the Project Browser displays look like this:



The user will decide where to put the cloned element, and remove the package called "Cloned element" This package was created with the intention to clearly distinguish the original element of cloning, since they are apparently equal.

DISCONNECT BRANCH FOR VERSION CONTROL

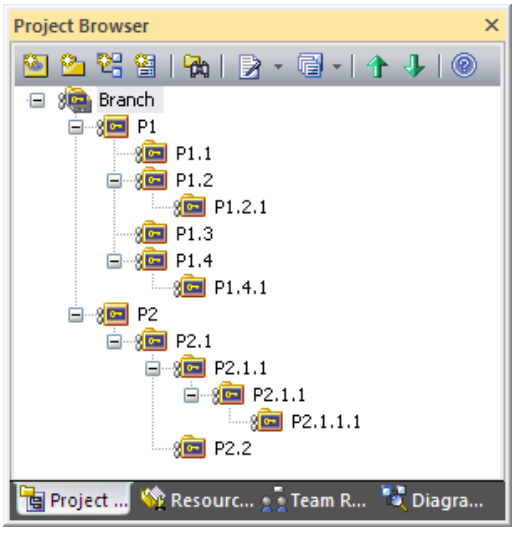
Enterprise Architect provides in the area of version control through external tools (Subversion, CVS, TFS, AccuRev, etc ...) some functions that operate on a massive scale against controlled packages. These functions are:

- Check-Out Branch
- Check-In Branch

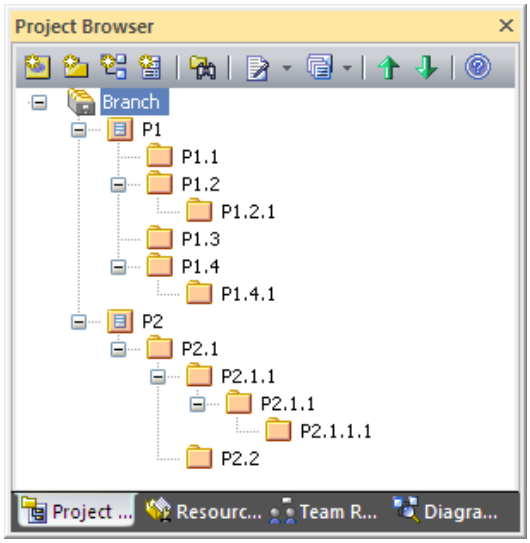
These functions allow in a single operation, carry out a Check-In Check-Out or an entire branch of packages. However, there is no function to disconnect a complete branch of version control, which means having to disconnect each package manually.

The "Disconnect Branch from VC" allows just that: removing the connection of an entire branch of version control packages automatically.

To use this function, simply select the package from which you want to start off and use the menu “EA Extension for QA | Disconnect Branch from VC” This function, applied to the "Branch" package of this example, work like this:



Would change the repository completely free from version control, it means, this function will remove the connection of each package of each branch from system configuration management:



CHANGE LANGUAGE

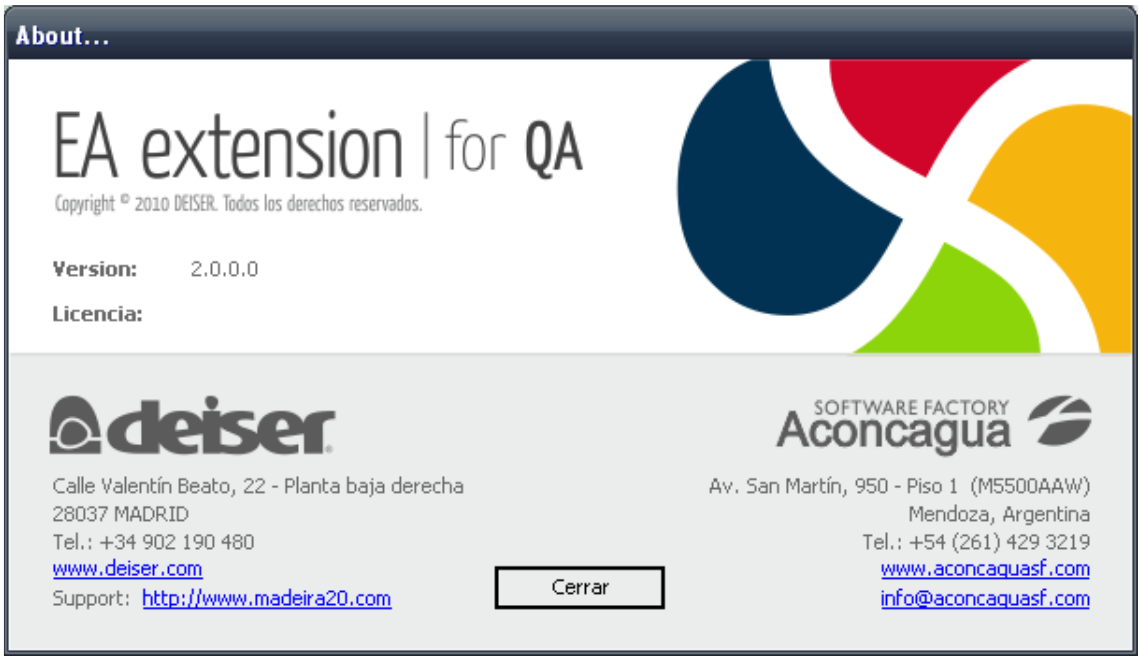
This function allows you to set the language of the user interface of the supplement. You can choose between Spanish and English.

ADD LICENSE NUMBER

This function will be available before you have activated the software with a valid license, and allows you to enter the license.

ABOUT...

This function displays information about the manufacturer of the supplement as well as the address of the support and active license if any.



deiser | Xpress Learning

C/ Valentín Beato, 22 - Planta baja derecha
28037 MADRID
Tel.: 902 190 480
Fax: 911 531 576
www.deiser.com